演化計算 Hw1

0416235 劉昱劭

1. The eight queens problem

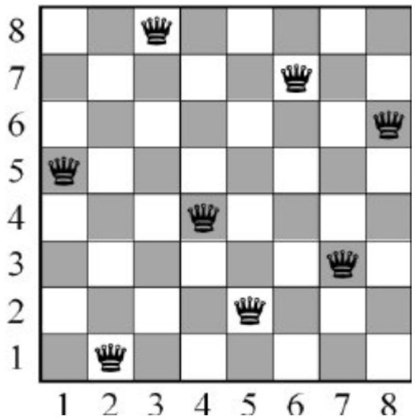
- (a) How big is the phenotype space for the eight queens problem?

> 可視為在棋盤中的 8 行，每行恰放一個皇后，
> 每列也恰有一個皇后，第 i 個放上棋盤的皇后有 9-i 列可選擇
> 即為：$8! = 8 \times 7 \times ... \times 2 \times 1 = 40320$ 種可能。
>
> (這樣要處理的 objective function 就可以當作是：每個皇后在兩個斜線方向上，所看到的其他皇后數量。)

- (b) Give a genotype to encode the 8x8 chessboard configuration.

> 用一個「有 8 個元素的陣列」。
> index 表第幾個 Column 有放皇后
> value 表第幾個 Row 有放皇后

例子 (假設 index 是 1-based)：



Column(index)	1	2	3	4	5	6	7	8
Row(value)	5	2	8	4	2	7	3	6

- (c) How big is the genotype space you give in 1b?

> 第 i 個元素有 9-i 種可能的值 (i=1,2,...,7,8)
> $8! = 8 \times 7 \times ... \times 2 \times 1 = 40320$ 種可能。

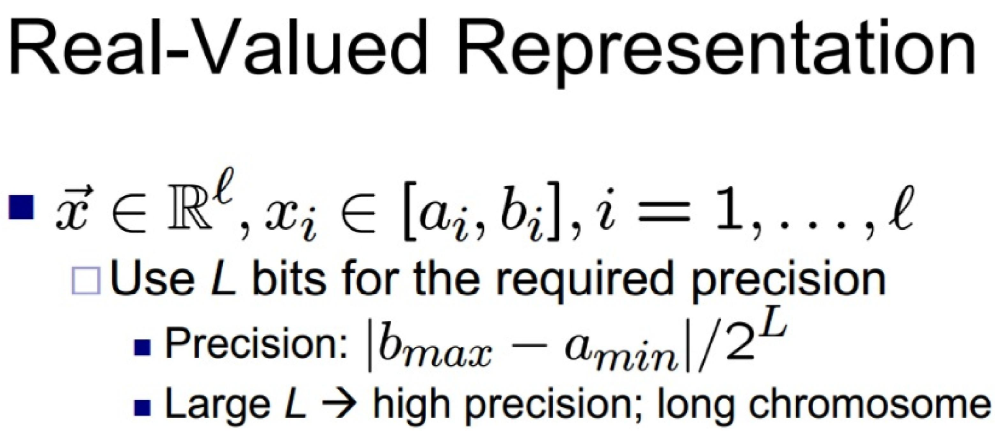
- (d) Briefly describe why the proposed genotype is able to cover the phenotype space.

> 這個 encode 方法是 phenotype 到 genotype 的一一對應。
> 每個皇后有其座標 (column, row)，可由陣列的 index 和 value 表示，
> index 表第幾個 Column 有放皇后
> value 表第幾個 Row 有放皇后

2. Precision problem

> How many bits are needed at least to
> achieve this precision (≤ 0.001) for a bit-string genetic algorithm?

- 參考 投影片 04 p28

> 
>

  - 對於 $x \in [0, 1]$，我們可以用 L 個 bit 的 bit-string 來表示，可記為：($x_i \in \{0, 1\}, i = 1, 2, \ldots, L$)
  - 這種表示法的精準度即為 $precision(L) = \dfrac{1-0}{2^L}$
  - 題目要求 $precision(L) = \dfrac{1-0}{2^L} \leq 0.001$
    $10^3 \leq 2^L$
    $L \geq 10$
  - 即最少要 **10 bits** 才能保證精準度到達 0.001

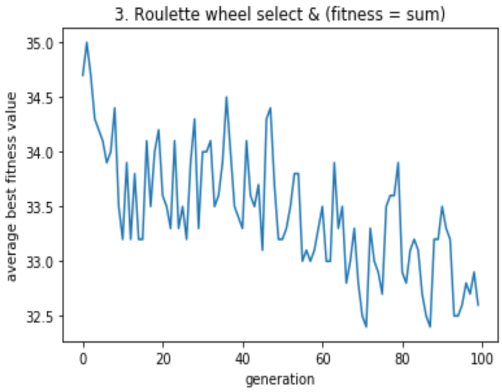
3~9 50-bits OneMax problem

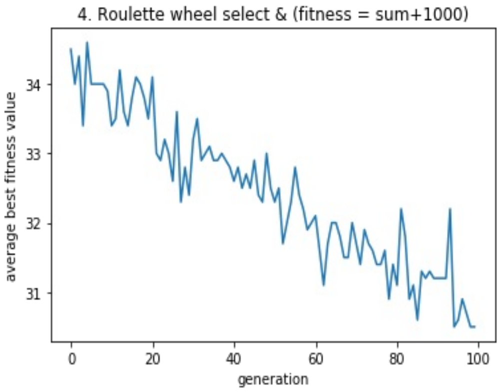
> • Fitness function:
> $$f(x) = \sum_{i=1}^{50} x_i + shift, \quad x_i \in \{0, 1\}, \quad i = 1, \ldots, 50$$
> $shift = \{0, 1000\}$
> • Binary representation;
> • Random initialization;
> • Parent selection: Roulette wheel selection with replacement;
> • Recombination: One-point crossover with probability pc = 1.0;
> • No mutation operator;
> • Replacement: Generational model (no elitism);
> • Population size: 200;
> • Termination criterion: 100 generations.

3. Roulette wheel & fitness = sum + 0 (shift=0)

4. Roulette wheel & fitness = sum + 1000 (shift=1000)

5. cf. 3, 4



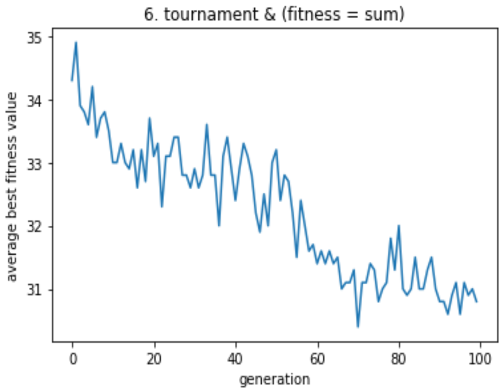


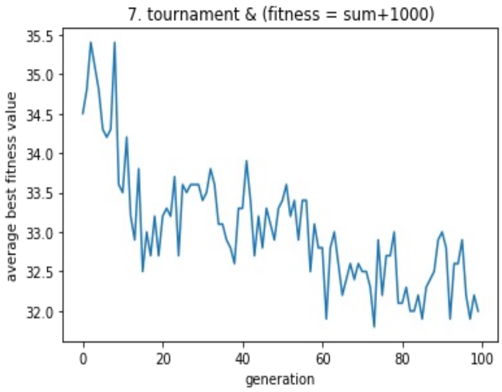
- (先以第 3. 題的 fitness function 討論)
  Roulette wheel selection 說 fitness value 越高的 individual，挑選他的機率密度越大，我們考慮以下兩個狀況：
  - f (x)=50，被挑到的機率是 $\frac{50}{\sum_{y=1}^{50} y} = 50/1275 \approx 0.04$
    - f (x)=50 的情況有 $C_{50}^{50} = 1$ 種可能
  - f (x)=25，被挑到的機率是 $\frac{25}{\sum_{y=1}^{50} y} = 25/1275 \approx 0.02$
    - f (x)=25 的情況有 $C_{25}^{50} \approx 1.2 \times 10^{14}$ 種可能
  - 由上可知，每個 individual 被挑到的機率是：
    「出現在 pool 內的次數」×「轉輪法給他的機率密度」。
  - 結果：
    越靠近平均值的 parent 在 pool 中出現次數應該會越多，**出現次數**會遠超過**機率密度**的影響，因此 fitness 高的不一定容易被挑到， search space 的上界會越來越往平均值靠近 (e.g. 0~100 --> 0~50)，所以 best fitness value 會越來越低。
- 第 4. 題又把 fitness value 加 1000，我們考慮上面兩種情況：
  - f (x)=50+1000，被挑到的機率是 $\frac{50+1000}{\sum_{y=1}^{50} (y+1000)} = 1050/51275 \approx 0.02$
    - f (x)=1050 的情況有 $C_{50}^{50} = 1$ 種可能
  - f (x)=25+1000，被挑到的機率是 $\frac{25+1000}{\sum_{y=1}^{50} (y+1000)} = 1025/51275 \approx 0.02$
    - f (x)=1025 的情況有 $C_{25}^{50} \approx 1.2 \times 10^{14}$ 種可能
  - 可以看到被挑到的機率都變成 0.02，因為 1000 遠大於左邊那項 sum (x)，因此左邊那項的影響幾乎可忽略，變成：
    $$\frac{\epsilon + 1000}{\sum_{y=1}^{50} (\epsilon + 1000)} \approx 1000/50000 = 0.02$$
  - 結果：讓 fitness 高的更難被挑到，也因此結果跑出來逼近期望值的速度比第 3. 題更快。

6. tournament & fitness = sum + 0 (shift=0)

7. tournament & fitness = sum + 1000 (shift=1000)

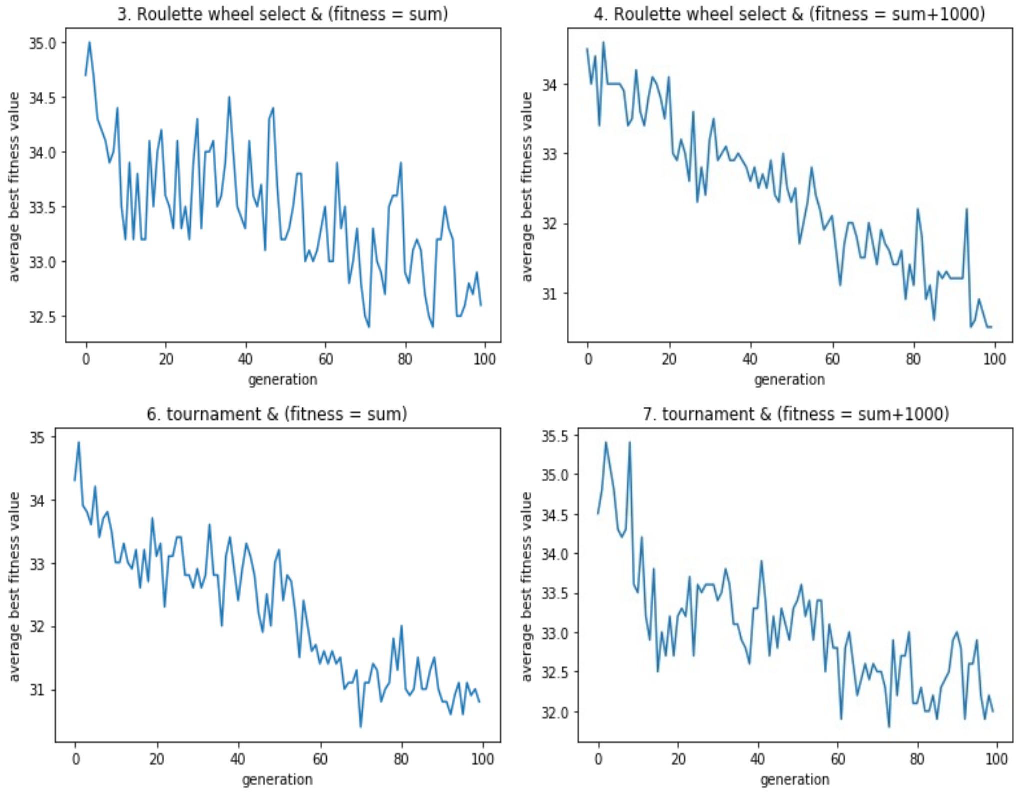
8. cf 6,7





- 對 tournament selection，這個 1000 的平移沒什麼意義，因為這個方法是 rank-based，每個 indiviual 的 fitness value 不管平移多少，他們相對的 rank 還是不變。
- 所以 6. 7. 沒什麼差異。

9. cf. 3,4,6,7



結果差不多，最優的個體 fitness 越來越低，主要原因應該有三個

- parent selection 容易挑出 fitness 靠近中間值的個體，長期來看 search space 容易縮小，且 search space 期望值容易下降。
- crossover 增加菁英的速度跟量都太慢，使用 N-point crossover 也許會比較好。
- survivor selection 沒使用 elitism，整體 fitness value 的期望值會容易被比較差的個體往下拖。

- 若觀察 worst individual，fitness 也越來越高，應表示 search space 有往中間縮小的趨勢。